MICRAL S

MICROCOMPUTER HANDBOOK

réalisations
études
électroniques

zone d'activités de courtabœuf
avenue de scandinavie
91400 orsay - france
907 47 77 - télex : 24014 f

MICRAL S

MICROCOMPUTER HANDBOOK

Ref. G 01 - E
August, 1974

TABLE OF CONTENTS

I - INTRODUCTION

The MICRAL "S" is the third model of the micro-computer series produced by R2E. Like its predecessors N and G MICRAL, this new micro-computer is built around a micro-processor in LSI technology. The "S" MICRAL has been developed from the 8080 INTEL micro-processor and it is the most powerful model of the series.

Several years of experience in system designing have shown R2E the advantage of disposing of a basic unit, at low cost which is standardized as much as possible with a modular design and is easily integrated into systems. This is why R2E propose a complete product with regard to the hardware and software.

In order to facilitate the adaptation of the MICRAL computers to their environment, R2E developed a full range of input and output modules for industrial, analog data acquisition and classical peripheral coupling.

For this purpose, the input/output system of the MICRAL is extremely powerful and its modularity is very high.

The present notice describes the essential features of this new micro-computer.

II - STARTING UP OPERATION OF THE "S" PROCESSOR

To replace the heart of a computer, that is, the processor, without altering the system, raises three difficulties :

1 - FORMAT PROBLEMS

Although the two computers are 8 bits machines, the 8080 and the 8008 L.S.I. are interfaced in a different way (bus bidirectional or indirectional, address field of 14 and 16 bits, displacement of the input-output fields).

2 - PROCEDURE PROBLEMS

The procedures of exchange are different. It is necessary therefore, to carry out an automatism which emulates the control signals of the 8008 but with the following restrictions :

. Do not slow down the throughput of the 8080.
. Adapt the signals to the timing of the existing modules.
. This is a delicate operation because the 8080 L.S.I. does not provide its internal status, as the 8008 does, requiring a communication bus, the "Pluribus".

3 - EXTENSION PROBLEMS

We have decided to incorporate at the processor level, a number of perfections such as :

. Quartz oscilator clock
. Programmable real-time clock
. Teletype controller

All these perfections require only one board which is of the MICRAL format.

III - BOARD DESCRIPTION

The board can be divided into a number of functional blocks.

FIRST BLOCK : IT CONTAINS THE INTEL "CPU" AND ITS INPUT-OUTPUT CIRCUITS

It also contains the continuous voltage converter providing suitable power to the 8080 L.S.I., as well as the generator of the additional clock which is piloted by a quartz oscilator clock at 20 MH_z.

SECOND BLOCK : FIELD AND FORMAT ADAPTER BLOCK

It permits passage of the standard 8080 fields to the addressing fields "Pluribus".

It is useful to note that the "Pluribus" addressing fields are already different from the 8008 standard fields. This is obtained by multiplexers which have a very fast logic (shottky),

... / 4

and provide an output of 3 status. In addition, an Aithmetical and a Logical Unit ALU of MSI comparators and adders with a very high speed allow emulation of the real Pluribus addresses.

THIRD BLOCK : EMULATION OF INTERNAL SIGNALS

This block allows reconstitution of the internal timing of the 8080 by different countings and testing of the environment.

The reconstitution of these signals has permitted the anticipation of some procedures and also to start them much before the 8080 standard signals are ready.

It is essentially for this reason and for the recovered time fractions that the following block, emulator of the control signals, is able to generate the necessary signals with security functioning of the Pluribus and of its connected modules.

It must be pointed out that the logic used is a rapid one, series S (3ns/gate).

FOURTH BLOCK : EMULATOR OF THE PLURIBUS CONTROL SIGNALS

This block transforms the internal signals provided by the 3 block into Pluribus control signals.

FIFTH BLOCK : REAL-TIME BLOCK

This block includes two sub-assemblies.

a) The Interrupt System

This system handles 8 hierarchised interrupt levels.

It authorises the individuel enabling and disabling of each level thus permitting a dynamic management of the priority by the software.

It is identical to the standard of the MICRAL. Nevertheless, the O level is no more dedicated.

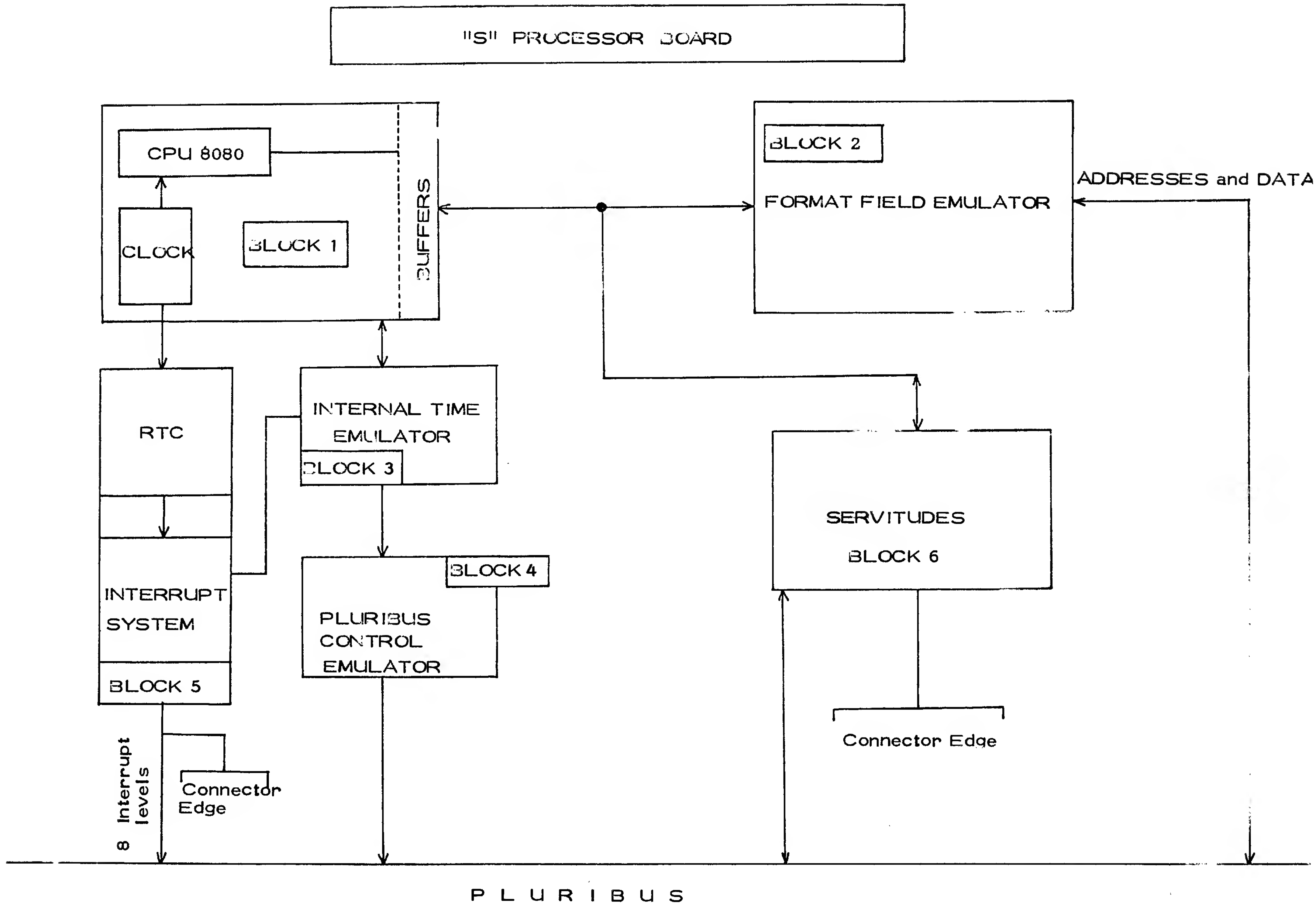
b) The Real-Time Clock

Obtained by the dimentioning of the basic clock, its precision and stability are 1×10^{-6}.

Moreover, this clock is program selectable (10 possible frequencies) and enabled or disabled by software.

SIXTH BLOCK : BOARD CONNECTION SYSTEM

This block contains specific logic such as input-output decoding which is necessary to the interrupt system and the programmable clock. It also contains the teletype controller and allows handling of the front panel switches and the watch dog feature.


"S" PROCESSOR BOARD
CPU 8080
CLOCK
BLOCK 1
BUFFERS
BLOCK 2
FORMAT FIELD EMULATOR
ADDRESSES and DATA
RTC
INTERNAL TIME EMULATOR
BLOCK 3
INTERRUPT SYSTEM
BLOCK 5
BLOCK 4
PLURIBUS CONTROL EMULATOR
SERVITUDES
BLOCK 6
Connector Edge
8 Interrupt levels
Connector Edge
P L U R I B U S

IV - SPECIFICATIONS

1 - HARDWARE

PROCESSOR BOARD DESCRIPTION

This board contains an INTEL micro-processor 8080 and its connected circuits which provide status decoding and synchronisation.

It also contains the following :

- Eight interrupt levels with priority system
- Programmable real-time clock
- Automatic restart feature
- A power-fail detection system
- A Pluribus generator allowing direct addressing up to 64K words of memory, and driving 248 input bytes, 96 output bytes and 8 channels
- A watch dog device providing illuminated indication

The 8080 is a C.P.U. with parallel processing on 8 bits. It is contained on a single L.S.I. chip with an N channel which provides the "S" MICRAL with much higher performances than the "N" (as much as 10 times faster).

The "S" MICRAL has been conceived in such a way that it is compatible with other MICRAL, namely N and G. Any program written for N or G models may be run on the "S" model. The "S" MICRAL benefits from all the hardware modules and environmental systems developed for the N and G models.

... / 8

Like the "N" MICRAL, the C.P.U. of the "S" MICRAL contains :

- An 8 bits Accumulator
- Six 8 bit registers
- Four condition codes
- An 8 bits A.L.U.

In addition, the 8080 has instructions for performing decimalized arithmetical operations and double length words (16 bits). This simplifies the addressing calculations and allows the carrying out of high-speed arithmetical operations.

The 8080 controls a stack in memory, located at an ordinary address. The stack has a LIFO (LAST IN-FIRST OUT) organisation, which safeguards and restores the accumulator, the condition indicators and the registers.

The 8080 contains a 16 bits internal stack pointer controlling the external stack. One of the main advantages of the stack structure is that it saves the program contexts at the occurance of any interrupt, and restores it at the end of the interrupt servicing sequence. Another advantage is that it permits an illimited number of subroutine calls.

The stack pointer is attained by the software.

The "S" MICRAL has 74 powerful instructions which raise the application fields of the "N" MICRAL. These instructions may be arranged in several groups as follows :

... / 9

- Register and memory transfer instructions
- Conditional or unconditional branches
- Conditional or unconditional subroutine calls
- Input-output instructions
- Save of the register contents
- Double length operation :

  . Immediate loading
  . Increment/decrement/add
  . Index register modification
  . Loading and storing of the H and L registers

- Indirect jump
- Modification of the stackpointer
- Logical operations
- Binary arithmetic
- Decimal arithmetic
- Memory or register increment/decrement
- Interrupt enabling and disabling
- Interrupt masking and unmasking
- Interrupt status check

The "S" MICRAL allows several direct or indirect, and immediate addressing modes by register.

The following chapters will describe the processor board of the "S" MICRAL.

REGISTERS

Seven registers are are directly addressable by the instructions

A	Accumulator
B C D E H L	Auxiliary registers which can also be used as address registers

The F register contains the condition flags (parity, zero, sign, overflow).

The registers (seven) have an internal address in the micro-processor which is respectively : 7, 0, 1, 2, 3, 4, and 5. Address 6 corresponds to the memory whose address is pointed by the H and L registers.

The H register contains the 8 most significant bits of the memory address i.e. the page number.

The L register contains the least significant bits of the memory address i.e. the byte number, module 256. This allows direct addressing of the whole memory up to 64 K bytes.

PROGRAM COUNTER

The program counter is placed in an internal register. This program counter is incremented, except in the case of interrupt, during the last stage of the instruction cycle.

In the event of a subroutine call, or at each interrupt request, the contents of the program counter (which is the return address), is automatically stored in the external LIFO stack to the address pointed by the stack pointer. At the time of the return instruction the program jumps back to the address saved in the LIFO stack.

The program counter (16 bits) is program accessible.

CONDITION FLAGS

The condition flags are simultaneously set or reset by each arithmetical or logical instruction. Four of them are directly checked.

OVERFLOW

This indicator is set at 1 when the accumulator overflows.
It is reset to zero by the logical instructions.
It is not affected by the increment and decrement instructions on the auxiliary registers (B, C, D, E, H, and L).

ZERO

This indicator is set to 1 when an arithmetical or logical operation or when a register increment leads to a null result.

SIGN

This indicator is set to 1 when an arithmetical or a logical operation or a register increment leads to a result whose highest weight bit is 1.

The word "sign" has been kept although this term is not strictly correct as far as the two's complement is concerned. In the latter, the most significant bit is taken as the sign bit.

PARITY

This indicator is set to 1 when an arithmetical or logical operation, or, when the register increment gives an even number of 1 bits.

These indicators can be tested by a certain number of instructions allowing branching of the program if the given condition is true or false. The CY5 indicator represents the overflow on the bit 3 of the accumulator on the instruction DAA and is automatically taken into account.

INTERRUPT SYSTEM

Each interrupt line corresponds to one level. Each level can be individually enabled or disabled and globally masked or unmasked. The enabling state can be tested by software.

When an interrupt is activated, the program is returned to one of the eight addresses of page O :

/0 - /8 - /10 - /18 - /20 - /28 -/30 -/38

When the addresses are preceded by the following sign : /, they are specified in hexadecimal.

Each address can be the start address of a subroutine which is composed of eight consecutive instructions. This is a maximum in order to prevent any overflow in the next subroutine.

If the subroutine exceeds 8 instructions, a jump instruction is required to permit reaching the second part of the subroutine.

... / 13

As soon as the interrupt request is aknowledged, there is an automatic masking of the level. The level nb7 can be connected to the real-time clock by a means of a strap.

In addition, the interrupt system comprises of the 8 aknowledge outputs.

REAL-TIME CLOCK

The real-time clock is set with programmable periods and it can be triggered in such a way that the first period, after its individuel enabling, is perfectly gauged. The first interrupt occurs directly after its nominal period.

The relative precision is 1.10^{-6}, the systematical error is 1µs. The selection of the period is made amongst 10 values which are : 1, 10, 20, 100 milliseconds; 1, 10, 100 seconds ; 1, 10 minutes ; 1 hour.

PERIPHERAL ADDRESSING

The direct addressing capacity of the peripherals is 248 bytes in input and 96 bytes in output. As each peripheral can transfer 1 byte, this represents 1 984 addressable bits in input and 766 addressable bits in output.

In addition, it can be used up to eight channels, each channel having a maximal transfer rate of 1 mega byte per second.

The extension of multiplexed inputs and outputs allows the system capacity to increase almost indefinitely.

... / 14

CONTROL

The "S" MICRAL instructions are from 1 to 3 bytes. Each instruction needs 1 to 5 machine or memory cycles in order to fetch and perform each instruction. The machine cycles are designated as follows : M_1, M_2, M_3, M_4, M_5.

Each machine cycle requires 3 to 5 stages : T_1, T_2. T_3, T_4, T_5. Each stage corresponds to a clock period. The machine clock which is crystal driven has a precision of : $1.\ 10^{-6}$.

The M_1 machine cycle is always a memory addressing cycle corresponding to the fetching phase of the instruction. This phase lasts 4 or 5 clock periods. The M_2, M_3. M_4, and M_5 cycles have generally a duration of 3 clock periods.

The necessary number of stages by instruction is a minimum of 4 for one instruction without memory reference, (This process is similar to those of the arithmetical instructions on the registers and the accumulator.) up to a maximum of 18 for the most complicated instructions (exchange of the H and L register contents with the contents of the successive memory locations pointed in the stack.

At the frequency of two mega hertz, all the instructions will be performed in an interval of 2 to 9 micro-seconds on condition that there is no loss of cycles which would result in using memory devices with a slow access time.

Figure 1 gives the timing of the stages, showing the running of any instruction.

During the action of T_1, the content of the program is placed on the address bus.

T_1 is always followed by the stage T_2. The signals : READY, HOLD, and HALT are tested during this stage.

If the READY signal is true, the T_3 cycle can progress, but if not, the processor is placed in a waiting status until the READY becomes true.

The READY allows the synchronisation between the processor and the memory devices having different time access or, with peripheral devices having variable access time.

During the action of the T_3, the data, which come from the memory, are transfered on to the instruction register (during M_1 only). The decoding instruction register generates the basic signals in order to control the internal transfers of the data, the timing, and the necessary number of machine cycles for performing the instruction.

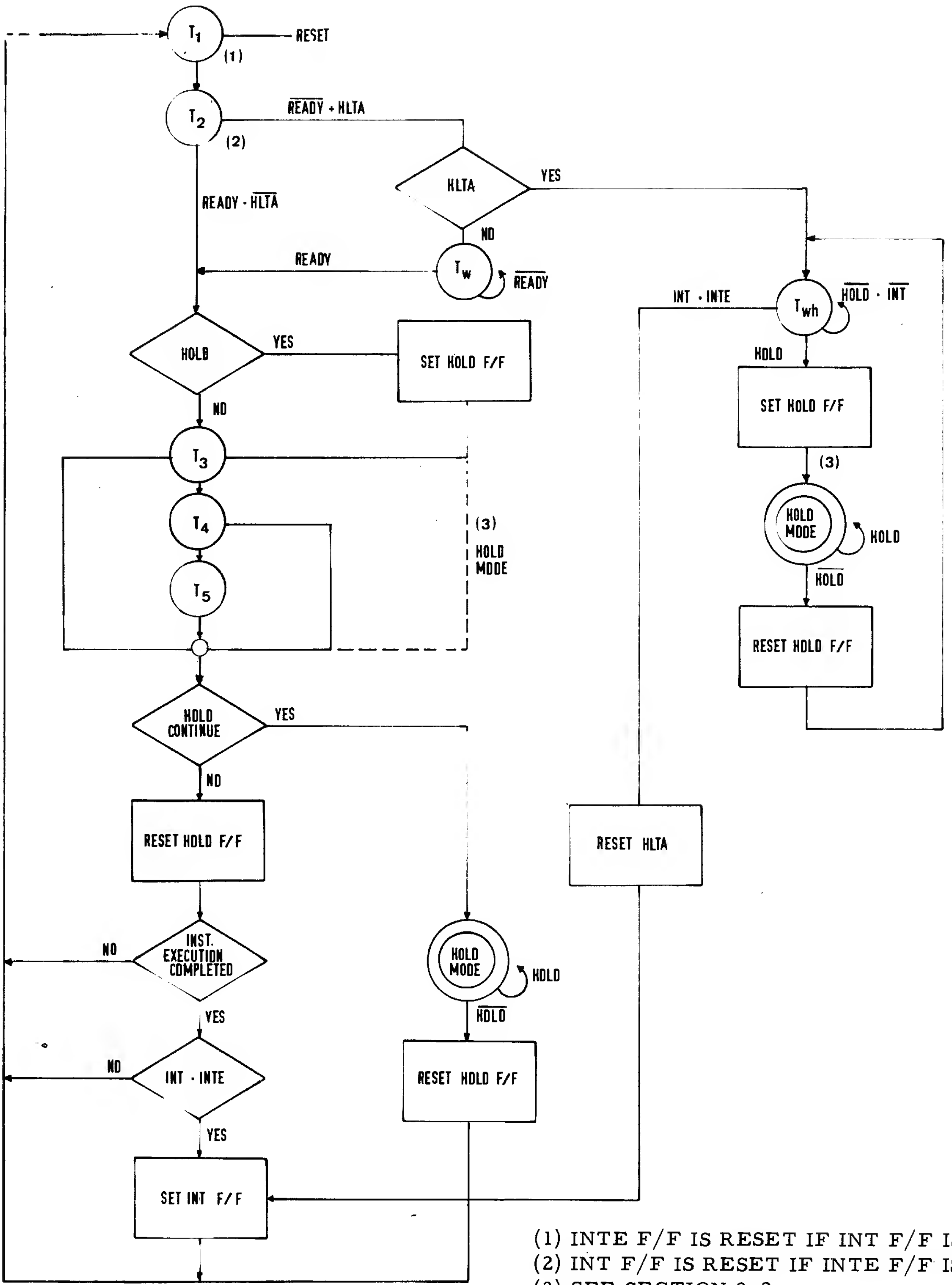
At the end of the T_4, (if the cycle is achieved, or not, on achievement of T_5) the 8080 returns to T_1 and begins execution of the M_2 machine cycle, unless the instruction requires only one machine cycle for its execution.

In the latter case, a new M_{1b} cycle is began. The loop is repeated until there is enough cycles and stages for instruction performing.

It is only during the last stage of the last machine cycle that the interrupt request line is tested and the special M_1 cycle is began. During this process, the program counter is not incremented and an interrupt status is returned. During this cycle, the address corresponding to the interrupt level (1 amongst 8) is returned to the C.P.U.

The required logic for controlling these interrupts is located on the processor board.

... / 16



(1) INTE F/F IS RESET IF INT F/F IS SET
(2) INT F/F IS RESET IF INTE F/F IS RESET
(3) SEE SECTION 3-2

Fig. 1 - STATE FLOW CHART

V - SOFTWARE

The "S" MICRAL, with its 8080 micro-processor provides for a high performing software simplifying the programming task. The main facilities are :

1 - EXTERNAL STACK

a) The presence of an external stack, which has a fairly illimited size avoids keeping in mind the risk of stack overflow.

b) The register save and restore instructions allow a fast preserving and restarting of the program content which is a major advantage for the interrupt handling.

2 - ADDRESSING

The "S" MICRAL offers four different ways of addressing a memory byte :

- By the H and L registers (the same method as in the "N" MICRAL)
- By the D and E registers (MDE and DEM instructions)
- By the B and C registers (MBC and BCM instructions)
- By an immediate addressing (AMI and MAI instructions)

THE "S" MICRAL START-UP

The whole set of programs written for the "N" MICRAL are available for the "S" MICRAL. All the instructions of the "N" MICRAL exist in the "S" MICRAL excepting those for the input-output which are changed. These instructions occupy 2 bytes in the S processor.

A written program for an "N" MICRAL will occupy the same memory size in the "S" MICRAL without using the new instructions.

THE "S" MICRAL INSTRUCTIONS

1 - DESCRIPTION OF THE SYMBOLES USED

<B2> Second instruction byte

<B3> Third instruction byte

R One of the reference registers A, B, C, D, E, H, L.

M The memory location indicated by the contents of registers H and L.

c One of the following flag flip-flop references :

carry	-	overflow, underflow
zero	-	result is zero
sign	-	MSB of result is "1"
parity	-	parity of result is even

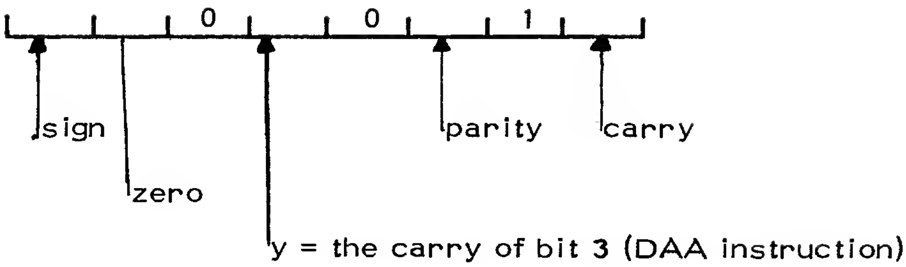
( ) Contents of location or register

$\leftarrow$ Transfer to

Am Bit m of the A-register

I Immediate operand indicator (for 2 byte instructions)

F The 5 flags represented in a byte under the following format :



[ ]	Indicates the element contained at the pointed address
PC	Program counter
SP	Stack pointer
K	8 bit literal
L	16 bit literal

2 - PRESENTATION OF EACH INSTRUCTION

- The condition flags are labeled in the following manner :

  c = carry z = zero s = sign p = parity

- Each letter means that the corresponding condition flag is modified by the instruction.

- A states represents 0, 5µs. with high speed memory which has an access time of $\leq$ 450 nano-seconds.

INSTRUCTION SET OF THE "S" MICRAL

1 - SINGLE BYTE LOAD

Mnemonic	Format	Description	Flags	Cycle No.	State No.
LR_1R_2	1	$(R_2) \rightarrow R_1$		1	5
LRM	1	(Memory) → R address given by (HL) registers		2	7
LMR	1	(R) → Memory address given by (HL)		2	7
LRI, K	2	K immediate → R		2	7
LMI, K	2	K immediate → Memory address given by (HL) registers		3	10

2 - DOUBLE BYTE LOAD

Mnemonic	Format	Description	Flags	Cycle No.	State No.
BCI, L	3	Literal (16 bits) → C, B Registers		3	10
DEI, L	3	L (16 bits) → D, E registers		3	10
HLI, L	3	L (16 bits) → L, H registers		3	10
SPI, L	3	L (16 bits) → Stack pointer		3	10

3 - ACCUMULATOR TRANSFER

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
MAI, AD	3	(A) → Memory location AD		4	13
AMI, AD	3	(Memory) → A		4	13
MBC	1	(A) → Memory location given by (B, C)		2	7
BCM	1	(Memory) → A, address given by (B, C) registers		2	7
MDE	1	(A) → Memory location given by (D, E) register		2	7
DEM	1	(Memory) → A, address given by (D, E) registers		2	7

4 - DIRECT LOADING OF THE (H) and (L) REGISTERS

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
MLH, AD	3	(L, H) → Memory location AD and AD+1		5	16
HLM, AD	3	(Memory) location AD → L (Memory) location AD+1 → L		5	16

5 - SINGLE BYTE INCREMENT

Mnemonic	Format	Description	Flags	Cycle No.	State No.
INR	1	(R) + 1 → R	zsp	1	5
INM	1	(Memory) + 1 → memory location given by H, L	zsp	3	11

6 - DOUBLE BYTE INCREMENTS

Mnemonic	Format	Description	Flags	Cycle No.	State No.
IBC	1	(B, C) + 1 → B, C registers	zsp	1	5
IDE	1	(D, E) + 1 → D, E registers	zsp	1	5
IHL	1	(H, L) + 1 → H, L registers	zsp	1	5
ISP	1	(SP) + 1 → SP	zsp	1	5

7 - SINGLE BYTE DECREMENTS

Mnemonic	Format	Description	Flags	Cycle No.	State No.
DCR	1	(R) - 1 → R	zsp	1	5
DCM	1	(Memory) + 1 → Memory location given by H, L	zsp	3	11

8 - DOUBLE BYTE DECREMENT

Mnemonic	Format	Description	Flags	Cycle No.	Stare No.
DCB	1	(B, C) - 1 ⟶ B, C registers	zsp	1	5
DDE	1	(D, E) - 1 ⟶ D, E registers	zsp	1	5
DHL	1	(H, L) - 1 ⟶ H, L registers	zsp	1	5
DSP	1	(SP) - 1 ⟶ SP register	zsp	1	5

9 - ARITHMETICAL OPERATIONS

The "A" Register is the Accumulator

- Add

Mnemonic	Format	Description	Flags	Cycle No.	State No.
ADR	1	(A) + (R) ⟶ A register	czsp	1	4
ADM	1	(A) + Memory ⟶ D	czsp	1	7
ACR	1	(A) + 1 + carry ⟶ A register	czsp	1	7
ACM	1	(A) + (Memory) + carry ⟶ A register. M location given by H, L registers	czsp	2	7
ACI, C	2	(A) +immediate operand + carry ⟶ A register	czsp	2	7

9 – Continued ......

– Substract

Mnemonic	Format	Description	Flags	Cycle No.	State No.
SUR	1	(A) – (R) ⟶ A register	czsp	1	7
SUM	1	(A) – Memory ⟶ A register location given by H, L register	czsp	1	7
SUI	2	(A) – immediate constant ⟶ A	czsp	2	7
SBR	1	(A) – (R) – carry ⟶ A	czsp	1	4
SBM	1	(A) – Memory – carry ⟶ A location given by H, L registers	czsp	2	7
SBI, k	2	(A) – immediate constant – carry ⟶ A	czsp	2	7

– Double Byte Add : (H) and (L) registers acting as Accumulator

Mnemonic	Format	Description	Flags	Cycle No.	State No.
HBC	1	(H, L) + (B, C) ⟶ H, L registers	c	3	10
HDE	1	(H, L) + (D, E) ⟶ H, L registers	c	3	10
HHL	1	(H, L) + (H, L) ⟶ H, L registers	c	3	10
HSP	1	(H, L) + (SP) ⟶ H, L registers	c	3	10

10 - LOGICAL OPERATION S

- Logic and

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
NDR	1	(A) anded with (R) ⟶ A register	c = o zsp	1	4
NDM	1	(A) anded with Memory ⟶ A register. Location biven by H, L registers	c = o zsp	2	7
NDI, c	2	(A) anded with immediate constant ⟶ A register	c = o zsp	2	7

- Exclusive or

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
XRR	1	(A) ex-ored with (R) ⟶ A register	c = o zsp	1	4
XRM	1	(A) ex-ored with Memory ⟶ A register	c = o zsp	2	7
XRI, c	2	(A) ex-ored with immediate constant ⟶ A register	c = o zsp	2	7

10 - Continued ...

- Inclusive or

Mnemonic	For- mat	Description	Flags	Cycle No.	State No.
ORR	1	(A) ored with (R) ⟶ A register	c = o zsp	1	4
ORM	1	(A) ored with Memory ⟶ A register. Location given by H, L register	c = o zsp	2	7
ORI, k	2	(A) ored with the immediate constant ⟶ A register	c = o zsp	2	7

11 - COMPARE INSTRUCTIONS : The (A) Register is unaltered

Mnemonic	For- mat	Description	Flags	Cycle No.	State No.
CPR	1	(A) compared with (R) (A) - (R)	czsp	1	4
CPM	1	(A) compared with (Memory) location given by H, L registers	czsp	2	7

Results : zero flag if (A) = 2nd operand

carry flag = 1 if (A) less than 2nd operand

carry flag = 0 if (A) greater than 2nd operand

12 - CIRCULAR SHIFT

- Shift Left

Mnemonic	Format	Description	Flags	Cycle No.	State No.
RLC	1	Logical rotate A contents left one bit. A7 ⟶ Ao and into carry flag	c	1	4
RAL	1	Arithmetical rotate A contents left one bit. Carry ⟶ Ao A7 ⟶ carry	c	1	4

- Shift Right

Mnemonic	Format	Description	Flags	Cycle No.	State No.
RRC	1	Logical rotate A contents right one bit Ao ⟶ A7 and into carry flag	c	1	4
RAR	1	Arithmetical rotate A contents right one bit Carry flag ⟶ A7 Ao ⟶ carry flag	c	1	4

13 - JUMPS and BRANCHES

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
JMP, AD	3	Unconditional jump to location AD		3	10
JTC, AD	3	Conditional jump to location AD,		3	10
JTZ		If tested condition is true			
JTS		If not, no jump			
JTP		i.e. (PC) + 3 ⟶ PC			
JFC, AD	3	Conditional jump to location AD,		3	10
JFZ		If the tested condition is false			
JFS		If not, no jump -			
JFP		(PC) + 3 ⟶ PC			

14 - CALL OF SUBROUTINE

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
CAL, AD	3	Call of a subroutine located at the address given by AD (PC) ⟶ Stack, address (SP) - 1 and (SP) - 2 (SP) - 2 ⟶ SP AD ⟶ PC		5	17

.../29

14 - Continued ...

Mnemonic	Format	Description	Flags	Cycle No.	State No.
CTC, AD	3	Conditional call of a subroutine		3/5	11/17
CTZ		If the tested condition is true			
CTS		Same characteristics as CAL			
CTP		If not, no call. (PC) + 3 $\longrightarrow$ PC			
CFC, AD	3	Conditional call of a subroutine		3/5	11/17
CFZ		If the tested condition is false			
CFS		Same characteristics as CAL			
CFP		If not, no call. (PC) + 3 $\longrightarrow$ PC			
RST, n	1	Call of Interrupt subroutine n = interrupt level, number between 0 and 7		3	11
		PC stored in stack, address (SP) - 1 qnd (SP) - 2			
		(SP) - 2 $\longrightarrow$ SP			
		(PC)= n multiplied by 8			

15 - SUBROUTINE EXIT

Mnemonic	Format	Description	Flags	Cycle No.	State No.
RET	1	Unconditional return The address pointed in the stack by (SP) and (SP) + 1 takes place into the PC (SP)+ 2 ⟶ SP		3	10
RTC RTZ RTS RTP	1	Conditional return If the tested condition is true acts az RET If not, no return		1/3	5/11
RFC RFZ RFS RFP	1	Conditional return If the tested condition is false acts as RET If not, no return		1/3	5/11

16 - INPUT/OUTPUT

Mnemonic	Format	Description	Flags	Cycle No.	State No.
INP, n	2	Input Data Byte ⟶ A n is the input byte number		3	10
OUT, n	2	A ⟶ Output Data Byte n is the output byte number		3	10

17 - REGISTER SAVE and RESTORE : (SP) = (SP) -2

Mnemonic	Format	Description	Flags	Cycle No.	State No.
SCB	1	Save C and B registers in the stack (B) ⟶ [SP - 1] (C) ⟶ [SP - 2]		3	11
SDE	1	Save D and E registers in the stack (D) ⟶ [SP - 1] (E) ⟶ [SP - 2]		3	11
SHL	1	Save H and L registers in the (H) ⟶ [SP - 1] (L) ⟶ [SP - 2]		3	11
SAF	1	Save A register and Flags in the stack (A) ⟶ [SP - 1] Condition Flags ⟶ [SP - 2]		3	11

18 - REGISTER STORE : (SP) = (SP) + 2

Mnemonic	Format	Description	Flags	Cycle No.	State No.
RBC	1	Restore B and C registers from the stack [SP] ⟶ C register [SP + 1] ⟶ B register		3	10
RDE	1	Restore D and E registers from the the stack [SP] ⟶ D register [SP + 1] ⟶ E register		3	10

.../32

18 - Continued ...

Mnemonic	Format	Description	Flags	Cycle No.	State No.
RHL	1	Restore H and L registers from the stack [SP] ⟶ H register [SP + 1] ⟶ L register		3	10
RAF	1	Restore A register and condition flags F [SP] ⟶ A [SP + 1] ⟶ Condition flags		3	10

19 - H AND L REGISTER HANDLING

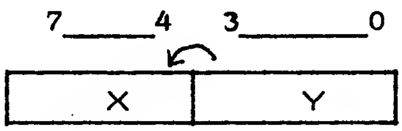
Mnemonic	Format	Description	Flags	Cycle No.	State No.
XDE	1	Exchange between D and E contents (H) ⟷ (D) (L) ⟷ (E) SP remains unaltered		1	4
XSK	1	Exchange between H, L registers, and stack (L) ⟷ [SP] SP remains unaltered		5	18
SPH	1	Store H, L contents in the stack pointer (H, L) ⟶ SP		1	5
PCH	1	Store H, L contents in the program counter (H, L) ⟶ PC Jump Indirect		1	5

20 - INTERRUPT CONTROL

Mnemonic	Format	Description	Flags	Cycle No.	State No.
DMS	1	Interrupt System Enable		1	4
MAS	1	Interrupt System Disable		1	4

21 - BCD FORMATTING

Mnemonic	Format	Description	Flags	Cycle No.	State No.

DAA	1	Decimal Adjust Accumulator
		The 8-bit value in the accumulator containing the result from an arithmetic operation on decimal operands is adjusted to contain two valid BCD digits by adding a value according to the following rules :
		
		Accumulator
		If ($Y \geq 10$) or (carry from bit 3) then $Y = Y + 6$ with carry to X digit.
		If ($X \geq 10$) or (carry from bit 7) or ($Y \geq 10$) and ($X = 9$) then $X = X + 6$ (which sets the carry flip-flop).
		Two carry flip-flops are used for this instruction. CY_1 represents the carry from bit 3 (the fourth bit) and is accessible as a fifth flag. CY_2 is the the carry from bit 7 and is the usual carry bit.
			czsp	1	4

.../34

22 - MISCELLANEOUS

Mnemonic	For-mat	Description	Flags	Cycle No.	State No.
BAR		$\overline{(A)} \longrightarrow (A)$ Accumulator two's complement			4
CAR	1	$(\overline{Carry}) \longrightarrow (Carry)$. Carry complement	c	1	4
SCY	1	$(Carry) \longrightarrow 1$. Carry is set to 1	c = 1	1	4
HLT		Processor stop with (PC) = (PC) + 1			7
NOP		Non effective operation $(A) \longrightarrow A$ $(PC) + 1 \longrightarrow PC$			4

MICRAL S

·	A	B	C	D	E	H	L	M	I ..
LA	7F	78	79	7A	7B	7C	7D	7E	3E
LB	47	40	41	42	43	44	45	46	06
LC	4F	48	49	4A	4B	4C	4D	4E	0E
LD	57	50	51	52	53	54	55	56	16
LE	5F	58	59	5A	5B	5C	5D	5E	1E
LH	67	60	61	62	63	64	65	66	26
LL	6F	68	69	6A	6B	6C	6D	6E	2E
LM	77	70	71	72	73	74	75	76	36
IN	3C	04	0C	14	1C	24	2C	34	« « «
DC	3D	05	0D	15	1D	25	2D	35	« « «
AD	87	80	81	82	83	84	85	86	C6
AC	8F	88	89	8A	8B	8C	8D	8E	CE
SU	97	90	91	92	93	94	95	96	D6
SB	9F	98	99	9A	9B	9C	9D	9E	DE
ND	A7	A0	A1	A2	A3	A4	A5	A6	E6
XR	AF	A8	A9	AA	AB	AC	AD	AE	EE
OR	B7	B0	B1	B2	B3	B4	B5	B6	F6
CP	BF	B8	B9	BA	BB	BC	BD	BE	FE

...	C	Z	S	P
JF.	D2	C2	F2	E2
JT.	DA	CA	FA	EA
...				
CF.	D4	C4	F4	E4
CT.	DC	CC	FC	EC
.				
RF.	D0	C0	F0	E0
RT.	D8	C8	F8	E8

RST ·	
0	C7
1	CF
2	D7
3	DF
4	E7
5	EF
6	F7
7	FF

·	RLC	07
·	RRC	0F
·	RAL	17
·	RAR	1F
·	RET	C9
...	JMP	C3
...	CAL	CD
.	NOP	00
.	HLT	76

.. Input	INP	DB
· Register Saves	SCB	C5
	SDE	D5
	SHL	E5
	SAF	F5
· Double Increment	IBC	03
	IDE	13
	IHL	23
	ISP	33
... Immediate Double Charge	BCI	01
	DEI	11
	HLI	21
	SPI	31
· (BC) --> A	BCM	0A
· (DE) --> A	DEM	1A
... (Imm) --> A	AMI	3A
· Exchange HL <--> DE	XDE	EB
· H,L --> SP	SPH	F9
... H,L ---> (Imm)	MHL	22
· IT authorization	DMS	FB
· compl. of A	BAR	2F
· 1 ----> carry	SCY	37

.. Output	OUT	D3
· Register Store	RBC	C1
	RDE	D1
	RHL	E1
	RAF	F1
· Double Decrement	DBC	0B
	DDE	1B
	DHL	2B
	DSP	3B
· Double Add on H, L	HBC	09
	HDE	19
	HHL	29
	HSP	39
· A --> (BC)	MBC	02
· A --> (DE)	MDE	12
... A --> (Imm)	MAI	32
· Exchange HL <--> SK	XSK	E3
· H,L --> PC jmp indirect	PCH	E9
... (Imm) ---> H,L	HLM	2A
· IT authorization	MAS	F3
· compl. of carry	CAR	3F
· decimal justification (y : report bit 3)	DAA	27

∴ = Byte No. of instruction

SP = Stack pointer

SK = Value in stack

F = Indicators S Z 0 y 0 P 1 C

(β) = Memory is addressed by β

PC = Ordinal counter